OAuth 2.0 and OpenID Connect

Fabian Hauck

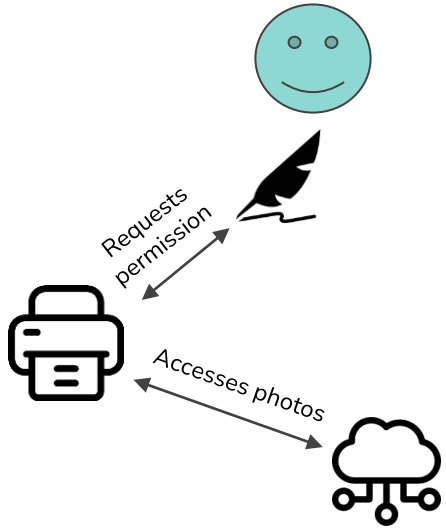
Overview

1. Introduction to OAuth 2.0 and OpenID Connect
2. Attacks on OAuth
3. Attack Mitigation
4. Outlook

OAuth 2.0 vs. OpenID Connect

OAuth 2.0: “The OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service [...]” - RFC 6749

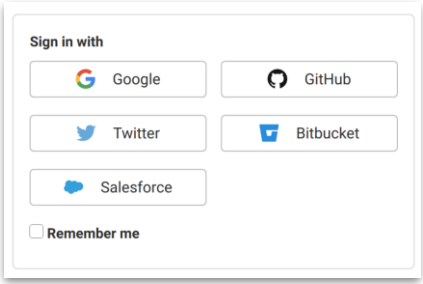
- Example use case: Authorize a printer to access a cloud storage with photos.



Icons made by Freepik from flaticon.com.

OpenID Connect: Is an identity layer on top of OAuth 2.0 that enables clients to verify the identity of a user. - OpenID Foundation

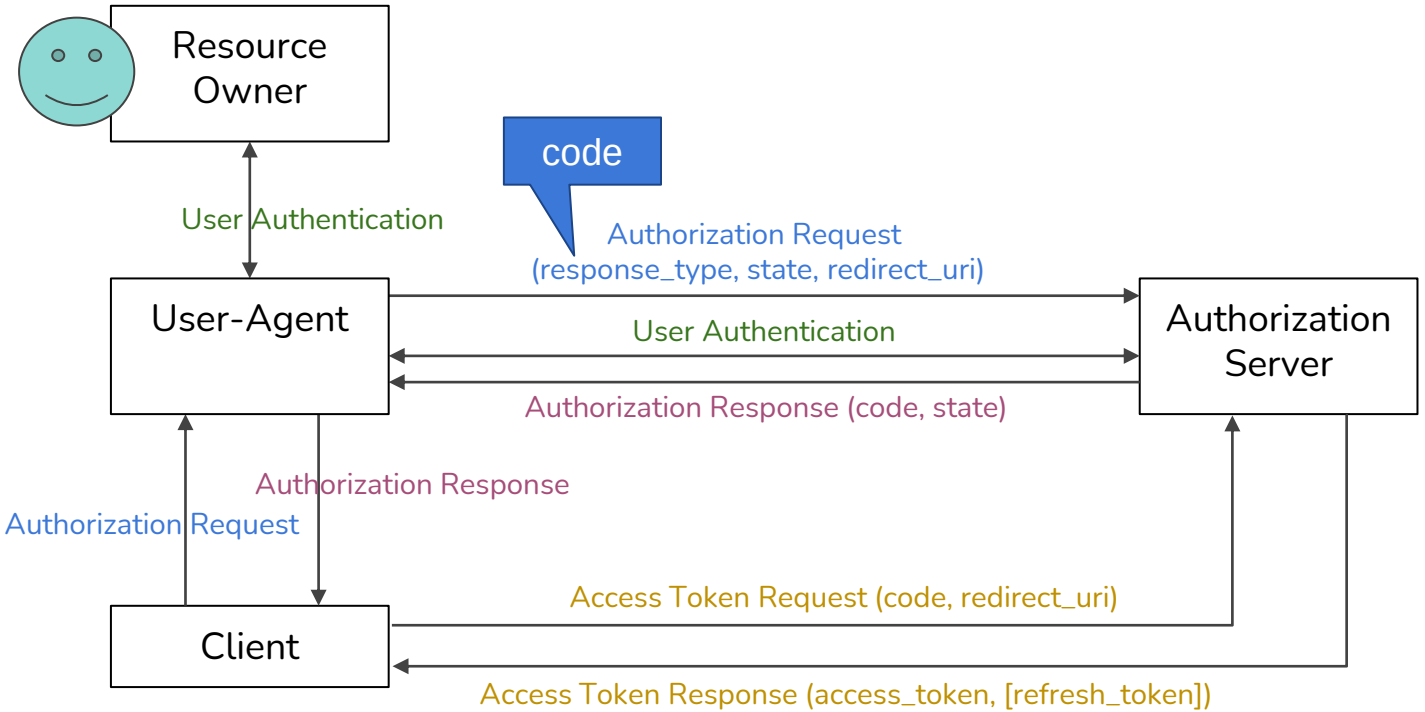
- Example use cases: Single Sign-On, Login to an application



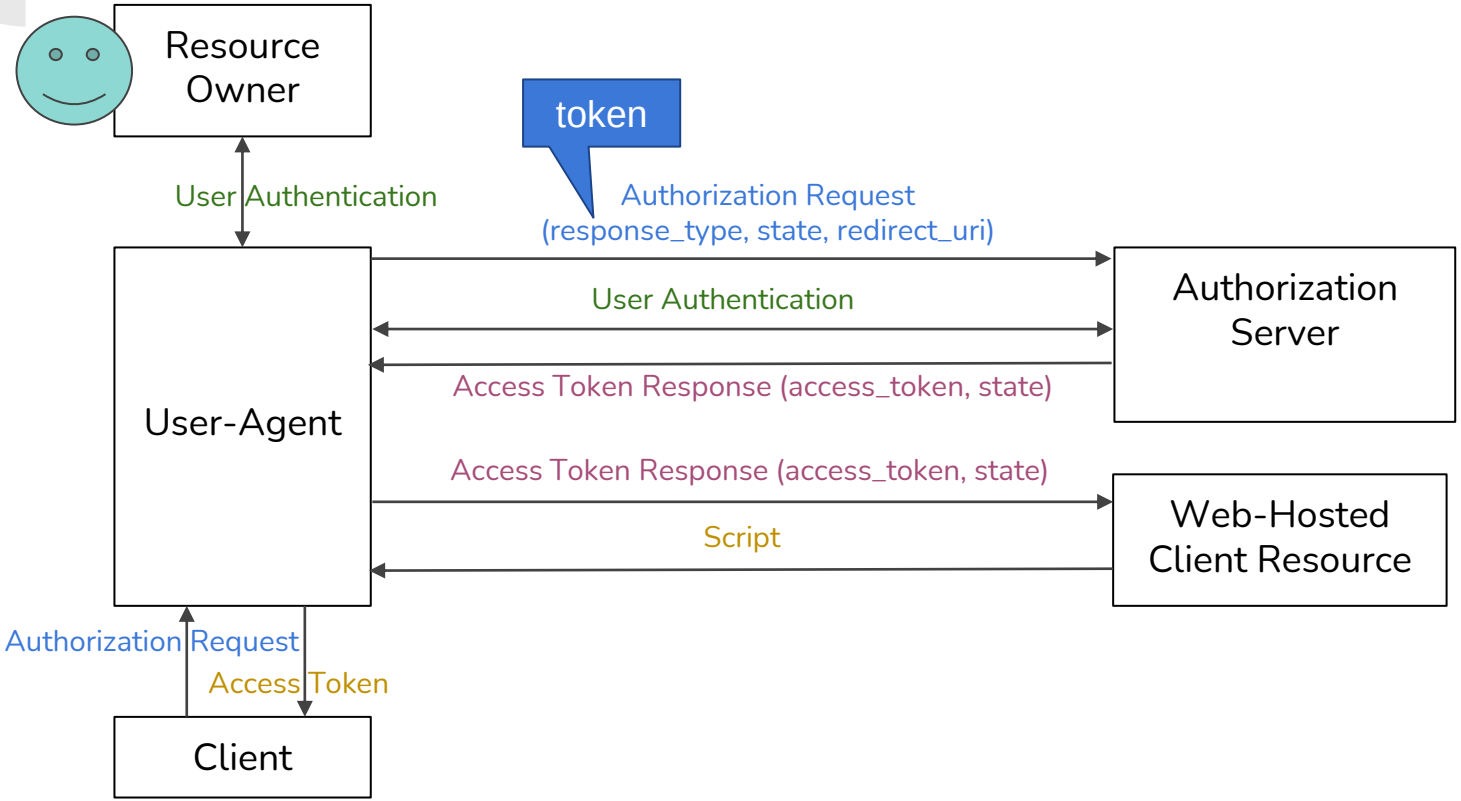
OAuth 2.0 Flows

- Authorization Code Grant
- Implicit Grant

Authorization Code Grant



Implicit Grant



Public vs. Confidential Clients

Confidential Clients:

- Can keep a client secret to authenticate to the authorization server
- Example: client implemented on a secure server

Public Clients:

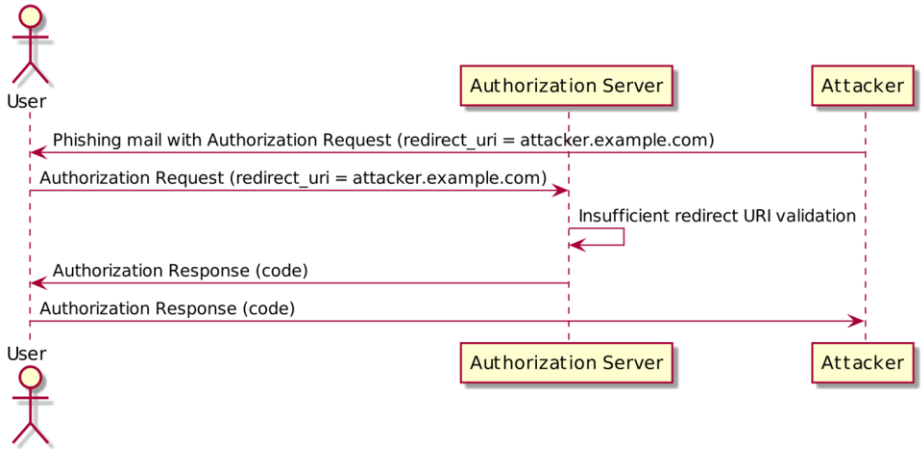
- Can not keep a client secret
- Example: native apps, web browser-based applications

Further information about client types can be found in RFC 6749.

Attacks

- **Insufficient Redirect URI Validation**
- **Misuse of Stolen Codes**

Insufficient Redirect URI Validation



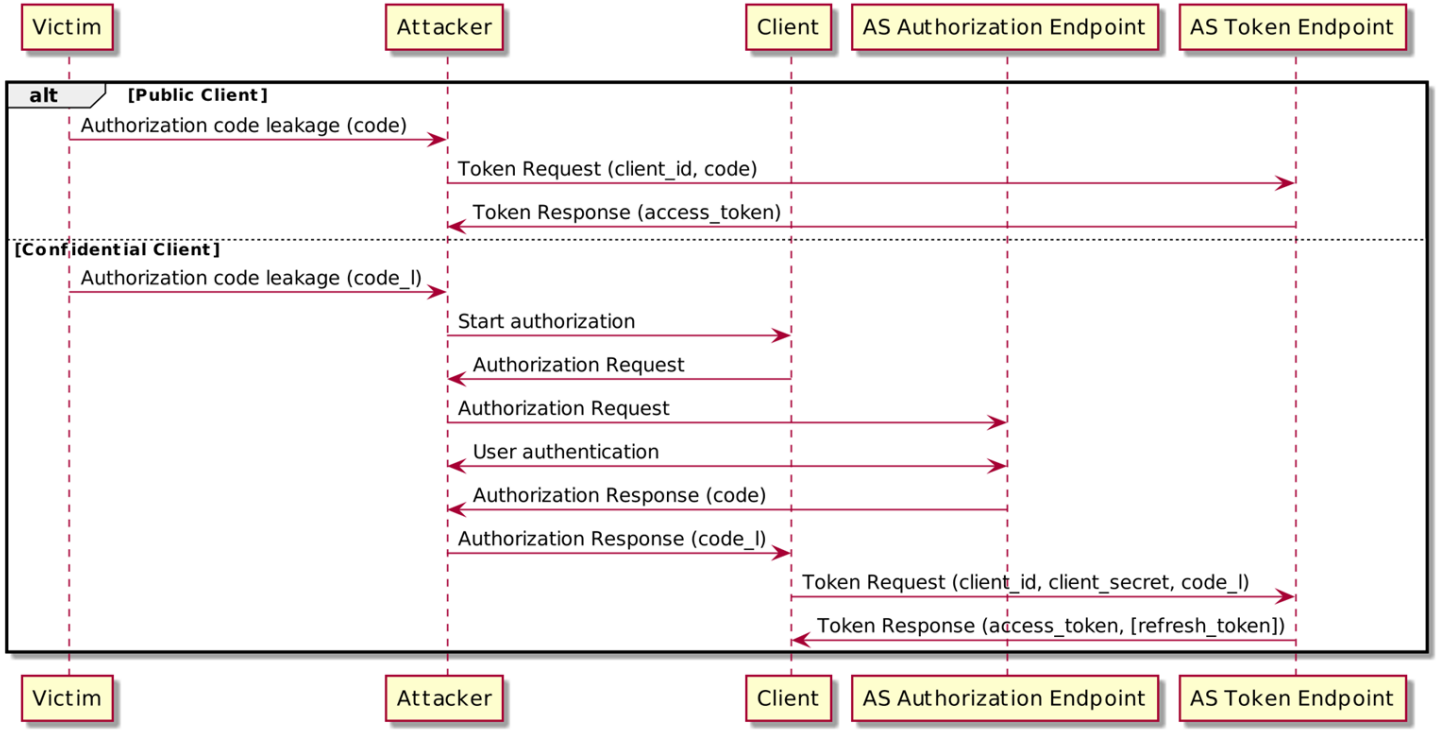
Attack on redirect URI wildcards:

- Redirect URI pattern: https://*.somesite.example/*
- Attackers redirect URI: https://attacker.example/.somesite.example

⇒ This could be a correct redirect URI depending on how the AS matches the wildcard.

Further information can be found in the OAuth 2.0 Security BCP.

Misuse of Stolen Codes



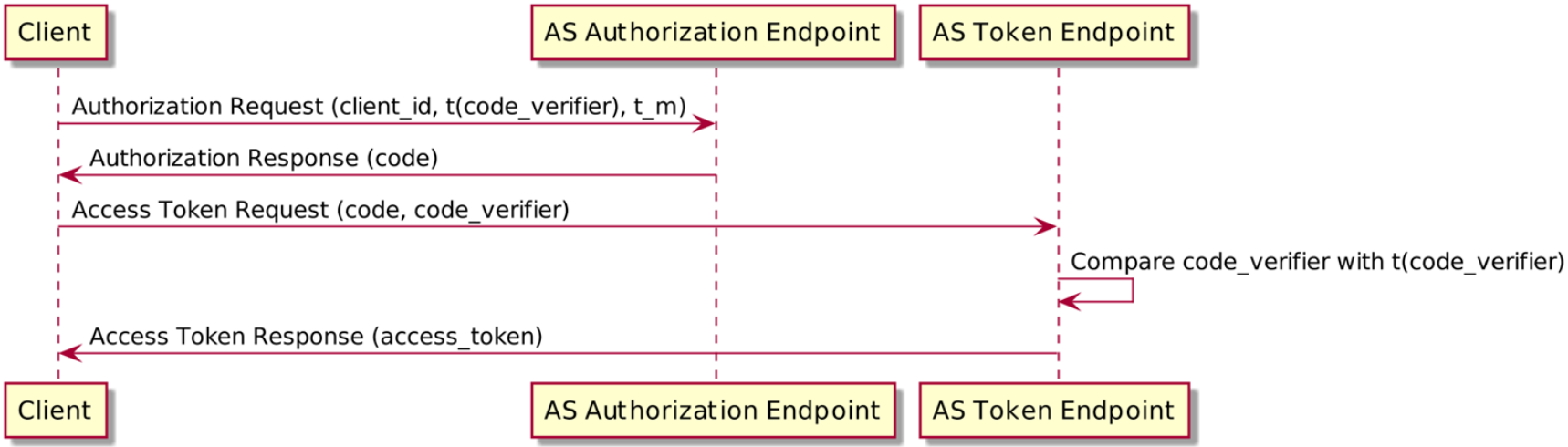
Further information can be found in this blog post by Daniel Fett and in the OAuth 2.0 Security BCP.

Security Mechanisms

- **PKCE**
- **Nonce**

Proof Key for Code Exchange (PKCE)

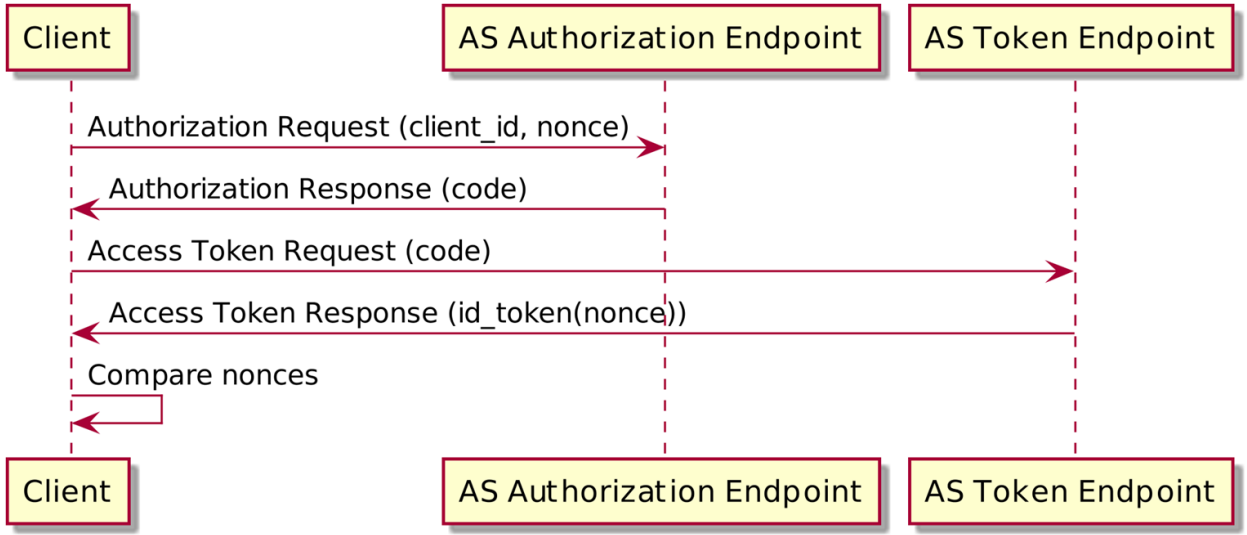
Security mechanism to protect the Authorization Code Grant against the authorization code interception attack.



Further information can be found in the RFC 7636.

Nonce (OpenID Connect)

Security mechanism to protect OIDC against replay attacks.

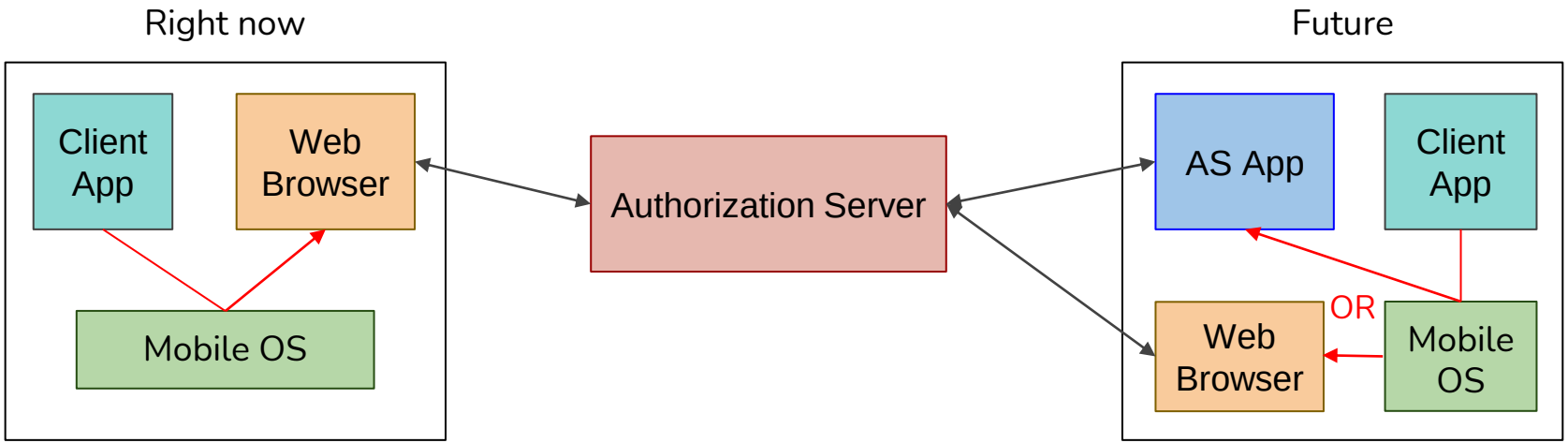


Further information can be found in the OpenID Connect Core specification.
For more information about PKCE vs. Nonce check out this blog post.

Outlook

App2App Authorization Flows

The next big thing in OAuth are App-to-App flows on mobile devices because they improve the user experience.



Further information can be found in the Improving OAuth App-to-App Security blog post.

OAuth 2.1

- The Authorization Code Grant has to use PKCE.
- Redirect URIs have to be compared by exact string matching.
- The Implicit flow and the Resource Owner Password Credentials flow were removed from the specification.
- Refresh tokens must either be bound to the client or refresh token rotation must be used.

Further information can be found in the OAuth 2.1 draft.

Live Demo

Demonstration of the authorization code interception attack on Android. The source code can be found on GitHub.

Guidelines for Penetration Testing

- Check if the appropriated flow is used (most times probably Authorization Code flow)
- Check for insufficient redirect URI verification
- Check whether PKCE and Nonce (OIDC) is used and verified correctly
- If no PKCE is used verify that the ‘state’ parameter is used and validated for CSRF protection → further information can be found in the OAuth 2.0 Security BCP
- Make sure that the authorization code cannot be reused
- Verify that all secrets (client_secret, state, nonce, pkce_verifier) have a sufficient high entropy and are not leaked through any channel
- On mobile: Check the security of the redirections (look at Improving OAuth App-to-App Security blog post)

Additional Resources

Penetration Tester's Guide to Evaluating OAuth 2.0 - Authorization Code Grant:

- https://maxfieldchen.com/posts/2020-05-17-penetration-testers-guide-oauth-2.html

OAuth to Account takeover:

- https://book.hacktricks.xyz/pentesting-web/oauth-to-account-takeover